AF

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s):	Russell C. Brown, Donald C. Likes, David A. Richardson, William A. Norris, Yurong Shi, Jeffrey B. Toth, Barry R. Hobbs		
Assignee:	Advanced Micro Devices, Inc.		
Title:	System for Integrating Data Between a Plurality of Software Applications in a Factory Environment		
Serial No.:	09/898,876	Filing Date:	July 3, 2001
Examiner:	Antony Nguyen-Ba	Group Art Unit:	2122
Docket No.:	TT3868	Customer No.:	33438

Austin, Texas
June 20, 2005

Mail Stop Appeal Brief - Patents
Board of Patent Appeals and Interferences
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF UNDER 37 CFR § 41.37

Dear Sir:

Applicant submits this Appeal Brief pursuant to the Notice of Appeal filed in this case on April 20, 2005. The Board is authorized to deduct the $500.00 Appeal Brief Fee from Deposit Account No. 01-0365 and to deduct any other amounts required for this appeal brief and to credit any amounts overpaid to Deposit Account No. 01-0365.

I. REAL PARTY IN INTEREST - 37 CFR § 41.37(c)(1)(i)

The real party in interest is the assignee, Advanced Micro Devices, Inc., as named in the caption above and as evidenced by the assignment set forth at Reel 011974, Frame 0283.

II. RELATED APPEALS AND INTERFERENCES - 37 CFR § 41.37(c)(1)(ii)

Based on information and belief, there are no appeals or interferences that could directly affect or be directly affected by or have a bearing on the decision by the Board of Patent Appeals and Interferences in the pending appeal.

III. STATUS OF CLAIMS - 37 CFR § 41.37(c)(1)(iii)

Claims 1 - 25 are pending in the application. Claims 1 - 25 stand rejected. The rejection of claims 1 - 25 is appealed. Appendix "A" contains the full set of pending claims.

IV. STATUS OF AMENDMENTS - 37 CFR § 41.37(c)(1)(iv)

No amendments after final have been requested or entered.

V. SUMMARY OF CLAIMED SUBJECT MATTER - 37 CFR § 41.37(c)(1)(v)

The present invention, as set forth by independent claim 1, relates to a system for integrating data between a plurality of software applications in a factory environment 100. The system and the plurality of software applications are stored on computer readable media. The system includes a factory system and a domain application. The factory system includes a domain object superclass 210, a plurality of first-level subclasses of the domain object superclass (e.g., 310, 320, 330, 340, 350, 360), an instantiation of one of the plurality of first-level subclasses corresponds to a domain object, the domain object represents an item in a factory, and a service. The service 120, 250 provides an operation related to the domain object. The service 120, 250 comprises at least one component. Each of the at least one component is operable to perform the operation related to the domain object 210. The domain application includes an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object 210.

The present invention, as set forth by independent claim 14, relates to a factory system for integrating data between a plurality of software applications in a factory environment 100. One of the plurality of software applications corresponds to a domain application. The factory system includes a domain object superclass 210, a plurality of first-level subclasses of the domain object superclass (e.g., 310, 320, 330, 340, 350, 360), an instantiation of one of the first-level subclasses of the plurality of first-level subclasses corresponds to a domain object, the domain object representing an item in a factory and a service 120, 250, the service 120, 250 providing an operation related to the domain object 210, the service comprising at least one component, each of the at least one component corresponding to operable to perform the

operation related to the domain object 210 wherein the domain application includes an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object 210.

The present invention, as set forth by independent claim 15, relates to a domain application for integrating data between a plurality of software applications in a factory environment 100. One of the plurality of software applications corresponds to a factory system. The factory system includes a domain object superclass 210, a plurality of first-level subclasses of the domain object superclass (e.g., 310, 320, 330, 340, 350, 360), an instantiation of one of the first-level subclasses corresponding to a domain object; the domain object represents an item in a factory and a service. The service 120, 250 provides an operation related to the domain object 210 using a component. The service 120, 250 comprises at least one component. Each of the at least one component is operable to perform the operation related to the domain object. The domain application includes an implementation of one component of the at least one component of the service of the factory system, wherein the component is operable to perform the operation related to the domain object 210.

The present invention, as set forth by independent claim 16, relates to a method for integrating data between a plurality of software applications in a factory environment 100 which includes providing a domain object superclass 210 in a first software application. The first software application corresponds to a factory system, providing a plurality of first-level subclasses of the domain object superclass (e.g, 310, 320, 330, 340, 350, 360), instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory, and providing a service 120, 250 that provides an operation related to the domain object, the service 120, 250 comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object, performing the operation related to the domain object using an implementation of one component of the at least one component of the service 120, 250, the implementation being provided by a second software application, the second software application corresponding to a domain application.

The present invention, as set forth by independent claim 21, relates to a computer program product for integrating data between a plurality of software applications in a factory environment 100 which includes instructions for providing a domain object superclass 210 in a first software application, the first software application corresponds to a factory system, instructions for providing a plurality of first-level subclasses of the domain object superclass (e.g., 310, 320, 330, 340, 350, 360), instructions for instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory, instructions for providing a service 120, 250 that provides an operation related to the domain object, the service 120, 250 comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object, and instructions for performing the operation related to the domain object using an implementation of one component of the at least one component of the service 120, 250, the implementation being provided by a second software application, the second software application corresponding to a domain application, and a computer-readable medium for storing the instructions for providing the domain object superclass 210, the instructions for providing the plurality of first-level subclasses, the instructions for instantiating, the instructions for providing the service, and the instructions for performing the operation.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL - 37 CFR § 41.37(c)(1)(vi)

Claims 1, 14 and 15 stand rejected under 35 U.S.C. 101.

Claim 16 stands rejected under 35 U.S.C. 101.

Claims 1, 14, 15, 16 and 21 stand rejected under 35 U.S.C. 102 over Baxter, U.S. Patent No. 6, 289,500 (Baxter).

VII. ARGUMENT - 37 CFR § 41.37(c)(1)(vii)

A. Claims 1, 14 and 15 are Directed to Statutory Subject Matter.

Claims 1, 14 and 15 stand rejected under 35 U.S.C. 101.

More specifically, with reference to claim 1, the Examiner sets forth:

> In claim 1, the limitation "system" is not clearly defined in the specification to be a system of software and hardware components. Therefore, claim 1 is now rejected under 35 U.S.C. § 112, first paragraph. Even assuming *arguendo*, that the specification does support this limitation, structure will not be read into claim 1 for the purposes of the statutory subject matter analysis. (Final Office action, Page 3.)

Claim 1 is directed to a "system". Claim 1 sets forth the structure of the system. Specifically, claim 1 sets forth that the system is "for integrating data between a plurality of software applications in a factory environment" and that the system and the plurality of software applications are "stored on computer readable media."

The Brown application sets forth

> In most factories it is desirable to have a single system that controls product flow and equipment usage. Such a system is referred to herein as a factory system. A factory system must overcome the complexities of communicating with tools that provide different types of data in varying formats. In most factories, the factory system interacts with numerous domain applications such as a manufacturing execution system, equipment interfaces, user authentication systems, statistical process control applications, engineering data analysis systems, equipment performance reporting applications, and material handling systems. (Brown, Page 2, lines 17 – 23.)

Additionally, the Brown application sets forth that "[p]hysical hardware and software are represented in the factory system as system objects." (Brown, Page 16, lines 22 – 24.)

When discussing patentable subject matter for computer related inventions, the MPEP sets forth

> The claimed invention as a whole must accomplish a practical result. . . . The purpose of this requirement is to limit patent protection to inventions that possess a certain level of "real world" value, as opposed to subject matter that represents nothing more than an idea or concepts, or is simply a starting point for future investigation or research. . . Accordingly, a complete disclosure should contain some indication of the practical application for the claimed invention, i.e., why the applicant believes the claimed invention is useful. (M.P.E.P. 2106A, *citations omitted.*)

Additionally, the Federal Circuit has addressed the issue of whether computer software is patentable subject matter in a number of decisions.

For example, In In re Alappat, the Federal Circuit set forth the view that certain types of mathematical subject matter, standing alone, represent nothing more than abstract ideas until

reduced to some type of practical application, and thus that subject matter is not, in and of itself, entitled to patent protection. For determining whether a claim is statutory subject matter, the focus must be on the claim as a whole. It is not necessary to determine whether a claim contains, as merely a part of the whole, any mathematical subject matter that standing alone would not be entitled to patent protection. In In re Alappat, the four claimed means elements functioned to transform one set of data to another through what may be viewed as a series of mathematical calculations. But that alone does not justify a holding that the claim as a whole is directed to nonstatutory subject matter. The claim is not so abstract and sweeping that it would wholly preempt the use of any apparatus employing the combination of mathematical calculations recited therein. In re Alappat, 33 F.3d 1526, 31 USPQ2d 1545 (Fed. Cir. 1994).

The Federal Circuit also addressed the issue of patentable subject matter in State Street Bank & Trust Co. V. Signature Fin Group Inc. (State Street). In State Street, the Federal Circuit sets forth:

> Today we hold that the transformation of data, representing discrete dollar amounts, by a machine through a series of mathematical calculations into a final price, constitutes a practical application of a mathematical algorithm, formula, or calculation, because if produces a 'a useful, concrete and tangible result'-a final share price momentarily fixed for recording and reporting purposes and even accepted and relied upon by regulatory authorities and in subsequent trades.
>
> The question of whether a claim encompasses statutory subject matter should not focus on which of the four categories of subject matter a claim is directed to -process, machine, manufacture, or composition of matter-but rather on the essential characteristics of the subject matter, in particular, its practical utility. Section 101 specifies that statutory subject matter must also satisfy the other 'conditions and requirements' of Title 35, including novelty, nonobviousness, and adequacy of disclosure and notice. See In re Warmerdam, 33 F.3d 1354, 1359, 31 USPQ2d 1754, 1757-58 (Fed. Cir. 1994). For purpose of our analysis, as noted above, claim 1 is directed to a machine programmed with the Hub and Spoke software and admittedly produces a 'useful, concrete, and tangible result.' Alappat, 33 F.3d at 1544, 31 USPQ2d at 1557. This renders it statutory subject matter, even if the useful result is expressed in numbers, such as price, profit, percentage, cost, or loss. State Street Bank & Trust Co. V. Signature Fin Group Inc., 149 F. 3d 1368, 47 USPQ2d 1596 (Fed. Cir. 1998).

The Federal Circuit also addressed the issue of whether software embodied on a computer readable media is statutory subject matter in In re Beauregard. The issue in In re Beauregard related to a Board rejection of computer program product claims on the grounds of a

printer matter objection. During the pendency of the appeal, the PTO commissioner issued an order that computer programs embodied in a tangible medium are patentable subject matter. Based upon the PTO commissioner's order, the Federal circuit vacated and remanded the appeal. (See e.g., In re Beauregard, 53 F. 3d 1583, 35 USPQ2d 1383 (Fed. Cir. 1995.)

Applicants respectfully submit that claim 1 is directed to the useful result of "integrating data between a plurality of software applications" and that claim 1 defines a useful machine as prescribed by identifying the physical structure of the machine. Applicants also respectfully submit that the invention as claimed is enabled under 35 U.S.C. __ 112, first paragraph.

With reference to claim 14, the Examiner sets forth:

> In claim 14, the "factory system" is described in the specification to be a single system that controls product flow and equipment usage (page 2, lines 17-18). As best understood by the Examiner, this factory system can be totally software-based (e.g., workflow software which is computer program per se) and as such is not a practical application that produces concrete and tangible results as required by State Street. Even assuming *arguendo*, that the factory system is a combination of software-hardware components, structure described in the specification will not be read into claim 14 for the purposes of the statutory subject matter analysis. (Final Office action, Page 3.)

Applicants respectfully submit that claim 14 is directed to the useful result of "integrating data between a plurality of software applications".

Claim 14 is directed to a "factory system for integrating data between a plurality of software applications in a factory environment". When discussing integrating data between software applications in a factory environment generally, the application sets forth

> data about the manufacturing process in the factory are sometime available as a live data stream from a tool in the factory, with content and format varying for each tool according to the vendor supplying the tool. Data about the manufacturing process are also available from domain applications, which collect data from tools and analyze the data. (Brown, page 2, lines 12 – 16.)

Claim 14 sets forth the factory system as well as a plurality of elements within the factory system (e.g., a domain object superclass, a plurality of first-level subclasses and a sevice). Accordingly, it is respectfully submitted that claim 14 defines a useful machine which produces

a tangible result that being integrating data between a plurality of software applications in a factory environment.

With reference to claim 15, the Examiner sets forth:

> In claim 15, the "domain application" is described in the specification at page 2, lines 25-30. As best understood by the Examiner, this domain application can be totally software-based (e.g., computer program per se) and as such is not a practical application that produces concrete and tangible results as required by State Street. Even assuming *arguendo*, that the domain application is a combination of software-hardware components, structure described in the specification will not be read into claim 15 for the purposes to the statutory subject matter analysis. (Final Office action, Pages 3, 4.)

Claim 15 is directed to a "domain application for integrating data between a plurality of software applications in a factory environment". Claim 15 sets forth the structure of the domain application as including an implementation of a component.

Accordingly, Applicants respectfully submit that, as with claim 14, claim 15 is directed to the concrete and tangible result of "integrating data between a plurality of software applications in a factory environment".

B. Claim 16 is directed to the useful result of "integrating data between a plurality of software applications".

Claims 16 stands rejected under 35 U.S.C. 101.

More specifically, the examiner has set forth

> Claims 16 is a method claim. The Examiner's interpretation of this claim is that it does not expressly require performance of any of the steps by a machine, such as a general purpose digital computer. Structure will not be read into the claim for the purposes of the statutory subject matter analysis even though the steps might be capable of being performed by a general purpose digital computer.
>
> Statutory subject matter requires two things: 1) it must be in the "technological arts;" and, if it is, 2) it must not fall within one of the exceptions for "laws of nature, physical phenomena and abstract ideas." Under the most recent Federal Circuit cases, transformation of the data by a machine (e.g., a computer) is statutory subject matter provided by the claims recite a "practical application, i.e., 'a useful, concrete and tangible result.'" State Street Bank & Trust v. Signature Financial Group, Inc., 149 F. 3d 1368, 1375 n. 9 (Fed. Cir. 1998). (Final Office action, Page 4.)

Claim 16 specifically relates to a "method for integrating data between a plurality of software applications in a factory environment." Claim 16 sets forth a plurality of elements within a system (e.g., a first software application corresponding to a factory system, a domain object representing an item in a factory, etc.). Accordingly, it is respectfully submitted that claim 16 inherently defines a useful machine which produces a tangible result that result being integrating data between a plurality of software applications in a factory environment.

C. Claims 1, 14, 15, 16 and 21 are allowable over Baxter, U.S. Patent No. 6, 289,500

Claims 1, 14, 15, 16 and 21 stand rejected over Baxter, U.S. Patent No. 6, 289,500 (Baxter).

The present invention, as set forth by independent claim 1, relates to a system for integrating data between a plurality of software applications in a factory environment. The system and the plurality of software applications are stored on computer readable media. The system includes a factory system and a domain application. The factory system includes a domain object superclass, a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the plurality of first-level subclasses corresponds to a domain object, the domain object represents an item in a factory, and a service. The service provides an operation related to the domain object. The service comprises at least one component. Each of the at least one component is operable to perform the operation related to the domain object. The domain application includes an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object.

The present invention, as set forth by independent claim 14, relates to a factory system for integrating data between a plurality of software applications in a factory environment. One of the plurality of software applications corresponds to a domain application. The factory system includes a domain object superclass, a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the first-level subclasses of the plurality of first-level subclasses corresponds to a domain object, the domain object representing an item in a factory and a service, the service providing an operation related to the domain object, the service

comprising at least one component, each of the at least one component corresponding to operable to perform the operation related to the domain object wherein the domain application includes an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object.

The present invention, as set forth by independent claim 15, relates to a domain application for integrating data between a plurality of software applications in a factory environment. One of the plurality of software applications corresponds to a factory system. The factory system includes a domain object superclass, a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the first-level subclasses corresponding to a domain object; the domain object represents an item in a factory and a service. The service provides an operation related to the domain object using a component. The service comprises at least one component. Each of the at least one component is operable to perform the operation related to the domain object. The domain application includes an implementation of one component of the at least one component of the service of the factory system, wherein the component is operable to perform the operation related to the domain object.

The present invention, as set forth by independent claim 16, relates to a method for integrating data between a plurality of software applications in a factory environment which includes providing a domain object superclass in a first software application. The first software application corresponds to a factory system, providing a plurality of first-level subclasses of the domain object superclass, instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory, and providing a service that provides an operation related to the domain object, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object, performing the operation related to the domain object using an implementation of one component of the at least one component of the service, the implementation being provided by a second software application, the second software application corresponding to a domain application.

The present invention, as set forth by independent claim 21, relates to a computer program product for integrating data between a plurality of software applications in a factory

environment which includes instructions for providing a domain object superclass in a first software application, the first software application corresponds to a factory system, instructions for providing a plurality of first-level subclasses of the domain object superclass, instructions for instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory, instructions for providing a service that provides an operation related to the domain object, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object, and instructions for performing the operation related to the domain object using an implementation of one component of the at least one component of the service, the implementation being provided by a second software application, the second software application corresponding to a domain application, and a computer-readable medium for storing the instructions for providing the domain object superclass, the instructions for providing the plurality of first-level subclasses, the instructions for instantiating, the instructions for providing the service, and the instructions for performing the operation.

Baxter discloses a mechanism for instantiating domain neutral objects with suitable domain specific run time extensions in an appropriate collection. Baxter further discloses as an example of an extension a domain specific extensible item factory. Baxter sets forth that the domain specific extensible item factory is a standard factory object that creates all extensible items and any owned extensions in the default collection. Before the object creates the extensible item, the object looks for a special factory to which the object can delegate the extensible item. Baxter sets forth that there is only need for one extensible item factory. (Baxter Col. 11, lines 1 – 5).

Baxter does not teach or suggest a system for integrating data between a plurality of software applications in a factory environment much less such a system which includes a factory system and a domain application where the factory system includes a domain object that represents an item in a factory and a service that provides an operation related to the domain object, all as required by claim 1. Accordingly, claim 1 is allowable over Baxter. Claims 2 – 13 depend from claim 1 and are allowable for at least this reason.

Baxter does not teach or suggest a factory system for integrating data between a plurality of software applications in a factory environment, much less such a factory system which includes a domain object superclass, a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the first-level subclasses of the plurality of first-level subclasses corresponds to a domain object, the domain object representing an item in a factory and a service, the service providing an operation related to the domain object, the service comprising at least one component, each of the at least one component corresponding to operable to perform the operation related to the domain object wherein the domain application includes an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object, all as required by claim 14. Accordingly, claim 14 is allowable over Baxter.

Baxter does not teach or suggest a domain application for integrating data between a plurality of software applications in a factory environment, one of the plurality of software applications corresponds to a factory system, the factory system includes a domain object superclass, a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the first-level subclasses corresponding to a domain object; the domain object represents an item in a factory and a service, the service provides an operation related to the domain object using a component, the service comprises at least one component and each of the at least one component is operable to perform the operation related to the domain object, much less such a domain application which includes an implementation of one component of the at least one component of the service of the factory system, wherein the component is operable to perform the operation related to the domain object, all as required by claim 15. Accordingly, claim 15 is allowable over Baxter.

Baxter does not teach or suggest a method for integrating data between a plurality of software applications in a factory environment which includes providing a domain object superclass in a first software application, the first software application corresponds to a factory system; providing a plurality of first-level subclasses of the domain object superclass; instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory; providing a service that provides an operation related to the domain object, the service comprising at least one component, each of the at least

one component being operable to perform the operation related to the domain object; and, performing the operation related to the domain object using an implementation of one component of the at least one component of the service, the implementation being provided by a second software application, the second software application corresponding to a domain application, all as required by claim 16. Accordingly, claim 16 is allowable over Baxter. Claims 17 – 20 depend from claim 16 and are allowable for at least this reason.

Baxter does not teach or suggest a computer program product for integrating data between a plurality of software applications in a factory environment which includes instructions for providing a domain object superclass in a first software application, the first software application corresponds to a factory system; instructions for providing a plurality of first-level subclasses of the domain object superclass; instructions for instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory; instructions for providing a service that provides an operation related to the domain object, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object; and instructions for performing the operation related to the domain object using an implementation of one component of the at least one component of the service, the implementation being provided by a second software application, the second software application corresponding to a domain application; and, a computer-readable medium for storing the instructions for providing the domain object superclass, the instructions for providing the plurality of first-level subclasses, the instructions for instantiating, the instructions for providing the service, and the instructions for performing the operation, all as required by claim 21. Accordingly, claim 21 is allowable over Baxter. Claims 22 – 25 depend from claim 21 and are allowable for at least this reason.

VIII. CLAIMS APPENDIX - 37 CFR § 41.37(c)(1)(viii)

A copy of the pending claims involved in the appeal is attached as Appendix A.

IX. RELATED PROCEEDINGS APPENDIX - 37 CFR § 41.37(c)(1)(x)

There are no related proceedings.

X. CONCLUSION

For the reasons set forth above, Applicant respectfully submits that the rejection of pending Claims 1 - 25 is unfounded, and requests that the rejection of claims 1 - 25 be reversed.

I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class Mail in an envelope addressed to: Mail Stop Appeal Brief – Patents, Board of Patent Appeals and Interferences, U.S. Patent and Trademark Office, P.O. Box 1450, Alexandria, VA 22313-1450, on June 20, 2005.

Attorney for Applicant — Date of Signature: 6/20/05

Respectfully submitted,

Stephen A. Terrile
Attorney for Applicant
Reg. No. 32,946

CLAIMS APPENDIX

1. A system for integrating data between a plurality of software applications in a factory environment, the system and the plurality of software applications being stored on computer readable media, the system comprising comprising:

a factory system comprising:

a domain object superclass;

a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the plurality of first-level subclasses corresponding to a domain object, the domain object representing an item in a factory; and

a service, the service providing an operation related to the domain object, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object;

and

a domain application comprising:

an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object.

2. The system of claim 1, wherein the domain application is one of a group consisting of the following:

a legacy system; and

an integrated application.

3. The system of claim 1 wherein

the domain application corresponds to an integrated application comprising:

a second-level subclass of one first-level subclass of the plurality of first-level subclasses of the domain object superclass of the factory system, an instantiation of the second-level subclass corresponding to an application-specific domain object;

and

the implementation of the one component corresponds to a method of the application-specific domain object, wherein the method is operable to perform the operation on the application-specific domain object, wherein the performing the operation on the application-specific domain object enables the domain object to communicate as if the operation were performed on the domain object.

4. The system of claim 1 wherein

the domain application corresponds to a legacy system comprising:

a data structure corresponding to the domain object;

and

the implementation of the one component corresponds to an interface to the legacy system, wherein the legacy system is operable to perform the operation on the data structure.

5. The system of claim 1, wherein

the operation comprises a plurality of operations.

6. The system of claim 1, wherein

the service comprises a plurality of services.

7. The system of claim 1, wherein

each component of the at least one component of the service has a corresponding domain application providing an implementation of the component.

8. The system of claim 1, wherein

the service includes instructions for selecting a component of the at least one component to perform the operation, the selecting providing a selected component; and

the selected component includes instructions to perform the operation.

9. The system of claim 1, wherein

a component of the at least one component is an interface to the domain application.

10. The system of claim 9, wherein

a requesting component of the at least one component includes instructions to use the interface to request the domain application to provide data to a receiving component of the at least one component; and

the receiving component includes instructions to receive the data from the domain application via the interface.

11. The system of claim 10 wherein

the receiving component and the requesting component are the same.

12. (Previously Presented) The system of claim 10 wherein

the receiving component further includes instructions to perform the operation on the domain object.

13. The system of claim 1 further comprising:

a system manager for managing hardware and software in the factory.

14. A factory system for integrating data between a plurality of software applications in a factory environment, one of the plurality of software applications corresponding to a domain application, the factory system comprising:

a domain object superclass;

a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the first-level subclasses of the plurality of first-level subclasses corresponding to a domain object, the domain object representing an item in a factory; and

a service, the service providing an operation related to the domain object, the service comprising at least one component, each of the at least one component corresponding to operable to perform the operation related to the domain object; and wherein

the domain application includes:

an implementation of one component of the at least one component of the service of the factory system to perform the operation related to the domain object.

15. A domain application for integrating data between a plurality of software applications in a factory environment, one of the plurality of software applications corresponding to a factory system, the factory system including: a domain object superclass; a plurality of first-level subclasses of the domain object superclass, an instantiation of one of the first-level subclasses corresponding to a domain object, the domain object representing an item in a factory; and a service, the service providing an operation related to the domain object using a component, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object; the domain application comprising:

an implementation of one component of the at least one component of the service of the factory system, wherein the component is operable to perform the operation related to the domain object.

16. A method for integrating data between a plurality of software applications in a factory environment comprising:

providing a domain object superclass in a first software application, the first software application corresponding to a factory system;

providing a plurality of first-level subclasses of the domain object superclass;

instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory; and

providing a service that provides an operation related to the domain object, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object;

performing the operation related to the domain object using an implementation of one component of the at least one component of the service, the implementation being provided by a second software application, the second software application corresponding to a domain application.

17. The method of claim 16 further comprising:

providing a second-level subclass of one first-level subclass of the plurality of first-level subclasses in the domain application, the domain application being an integrated application;

instantiating the second-level subclass to provide an application-specific domain object;

implementing one component of the at least one component as a method of the application-specific domain object;

performing the operation related to the domain object on the application-specific domain object using the method;

and wherein

the performing the operation related to the domain object on the application-specific domain object enables the domain object to communicate as if the operation were performed on the domain object.

18. The method of claim 16 further comprising:

providing a data structure corresponding to the domain object in the domain application, the domain application being legacy system;

implementing one component of the at least one component to serve as an interface to the legacy system;

requesting the legacy system to perform the operation related to the domain object via the interface; and

performing the operation related to the domain object on the data structure.

19. The method of claim 16 further comprising:

requesting the service to perform the operation related to the domain object;

selecting a selected component of the at least one component to perform the operation related to the domain object; and

performing the operation related to the domain object using the selected component.

20. The method of claim 16 further comprising:

requesting the domain application to provide data to a receiving component of the at least one component; and

receiving the data from the domain application by the receiving component.

21. A computer program product for integrating data between a plurality of software applications in a factory environment comprising:

instructions for providing a domain object superclass in a first software application, the first software application corresponding to a factory system;

instructions for providing a plurality of first-level subclasses of the domain object superclass;

instructions for instantiating one subclass of the plurality of first-level subclasses to create a domain object, the domain object representing an item in a factory;

instructions for providing a service that provides an operation related to the domain object, the service comprising at least one component, each of the at least one component being operable to perform the operation related to the domain object; and

instructions for performing the operation related to the domain object using an implementation of one component of the at least one component of the service, the implementation being provided by a second software application, the second software application corresponding to a domain application;

and

a computer-readable medium for storing the instructions for providing the domain object superclass, the instructions for providing the plurality of first-level subclasses, the instructions for instantiating, the instructions for providing the service, and the instructions for performing the operation.

22. The computer program product of claim 21 further comprising:

instructions for providing a second-level subclass of one first-level subclass of the plurality of first-level subclasses in the domain application, the domain application being an integrated application;

instructions for instantiating the second-level subclass to provide an application-specific domain object;

instructions for implementing one component of the at least one component as a method of the application-specific domain object;

instructions for performing the operation related to the domain object on the application-specific domain object using the method, wherein the performing the operation

enables the domain object to communicate as if the operation related to the domain object were performed on the domain object;

and wherein

the computer-readable medium further stores the instructions for providing the second-level subclass, the instructions for instantiation the second-level subclass; the instructions for implementing, and the instructions for performing the operation related to the domain object on the application-specific domain object.

23. The computer program product of claim 21 further comprising:

instructions for providing a data structure corresponding to the domain object in the domain application, the domain application being legacy system;

instructions for implementing one component of the at least one component to serve as an interface to the legacy system;

instructions for requesting the legacy system to perform the operation related to the domain object via the interface; and

instructions for performing the operation related to the domain object on the data structure;

and wherein

the computer-readable medium further stores the instructions for providing the data structure, the instructions for implementing the component to serve as the interface, the instructions for requesting, and the instructions for performing the operation related to the domain object on the data structure.

24. The computer program product of claim 21 further comprising:

instructions for requesting the service to perform the operation related to the domain object;

instructions for selecting a selected component of the at least one component to perform the operation related to the domain object; and

instructions for performing the operation related to the domain object using the selected component;

and wherein

the computer-readable medium further stores the instructions for requesting, the instructions for selecting a selected component, and the instructions for performing the operation related to the domain object using the selected component.

25. The computer program product of claim 21 further comprising:

instructions for requesting the domain application to provide data to a receiving component of the at least one component; and

instructions for receiving the data from the domain application by the receiving component;

and wherein

the computer-readable medium further stores the instructions for requesting and the instructions for receiving.